CS 579: Computational Complexity
Class Test

Wednesday, April 14, 2010

You have 75 minutes for the following six problems. The first five problems carry 10 points each (though some are easier/shorter than the others). If you have time, you can attempt the final extra credit problem (Problem 6). Any points you earn in this problem can make up for any points you lose in the other problems.

Problem 1. Define computation of a (possibly non-boolean) function by a non-deterministic TM as follows: an NTM M is said to compute a function f if on input x, all threads of execution which do not output `abort` output $f(x)$, and at least one thread does not output `abort`.

Consider a relaxation of polynomial time reduction, called *non-deterministic* polynomial time reduction. It is a many-one reduction, in which the reduction is computed (as defined above) by a polynomial time non-deterministic TM.

Show that if a language in **P** is **NP**-complete with respect to non-deterministic polynomial time reductions, then **NP** = **co-NP**.

Problem 2. Show that **polyL** := **DSPACE**$((\log n)^{O(1)})$ does not have any complete problems under log-space many-one reductions.

Problem 3. Show that **AM** $\subseteq$ **NP**/poly.

Problem 4. Show that if **IP** $\subseteq$ **P**/poly then **IP** = **MA**.

Hint: Recall that (the next bit function of) the prover in an IP protocol is a **PSPACE** *language, and that* **PSPACE** = **IP**.

Problem 5. We consider two restrictions to the class $\mathbf{P}^{\mathbf{NP}}$.

Let $\mathbf{P}_{||}^{\mathbf{NP}}$ denote the class of languages that can be decided by a deterministic polynomial time Turing machine, with access to an oracle for any **NP** language (say for SAT), *with the restriction that it must submit all its (polynomially many) queries to the oracle together* (that is it cannot base its second query on the answer to the first query and so forth).

Let $\mathbf{P}^{\mathbf{NP}[\log]}$ denote the class of languages that can be decided by a deterministic polynomial time Turing machine, with access to an oracle for any **NP** language (say for SAT), *with the restriction that it can query the oracle at most* $O(\log n)$ *times* (n being the input length).

1. Show that $\mathbf{P}^{\mathbf{NP}[\log]} \subseteq \mathbf{P}_{||}^{\mathbf{NP}}$.
2. Show that $\mathbf{P}_{||}^{\mathbf{NP}} = \mathbf{P}^{\mathbf{NP}[\log]}$.

   Hint: Suppose for the polynomially many parallel queries we could know exactly how many will be answered positively by the **NP** *oracle; given this number can we make a single query to SAT to check if the input will be accepted?*

Problem 6 (Extra Credit). Show that a language is computable by a polynomial-sized boolean formula[1] iff it is computable by a logarithmic depth, bounded fan-in boolean circuit (i.e., it is in non-uniform $\mathbf{NC}^1$).

Hint: Given a formula f with a "sub-formula" g, can you rewrite it so that the depth is no more than $\max(\text{depth}(f\backslash g), \text{depth}(g)) + 2$ *and size is no more than* $2(\text{size}(f\backslash g) + \text{size}(g))$*? ($f\backslash g$ refers to the formula obtained from f by replacing g by, say, a single new variable.)*

[1]Size of a formula refers to the number of literals appearing in it, as function of its input size. For e.g. size of $(x \vee y) \wedge (x \vee z)$ is 4, and its input size is 3.